Joint Eurographics-IEEE TCVG Symposium on Visualization, May 2000, to appear.

Integrated Multiresolution Geometry and Texture Models for Terrain Visualization

Konstantin Baumann, Jürgen Döllner, Klaus Hinrichs

Department of Computer Science, University of Münster
{kostab, dollner, khh}@uni-muenster.de

Abstract. In this paper, an approach for integrating multiresolution representations of terrain geometry and terrain texture data is presented. A terrain is modeled by a regular grid, which can be partially refined by local TINs in order to represent morphologically complex terrain parts. The multiresolution models for terrain texture data and geometry data are closely related: The rendering algorithm selects geometry and texture patches based on screen-space error criteria. Multiple texture hierarchies, which may represent different thematic information layers, can be bound to one terrain model. Multiple textures lead to a drastic improvement of visual quality: Topographic textures can be used to provide pixel-precise shading, alpha textures can be used to restrict or to highlight thematic textures. Multiple textures facilitate the development of visual interaction tools such as *magic lenses,* and texture animations. Multitexturing permits an efficient implementation of these concepts.

1 Introduction

In many kinds of virtual environments digital terrain models play a central role as fundamental tools to present and communicate spatial information. Various hierarchical data structures are suited for representing terrains, e.g., hierarchical TINs [6], R-trees [12], restricted quadtree triangulations [16], and progressive meshes [10]. Most multiresolution modeling schemes support a specific type of input data such as arbitrarily distributed data points for triangulated irregular networks (TINs), or regularly distributed data points for grids. In general, real-world terrain data sets are composed of data of different types. For example, a cartographic terrain model can include grid data describing the digital elevation model (DEM) and microstructures describing structures at a finer resolution such as topographically complex or interesting terrain parts (e.g., riverbeds illustrated in Fig. 1).

Texture data represent another important category of terrain data. Multiresolution modeling can be extended to texture data as well. In particular, for real-world terrain models, large, high-resolution textures need to be processed which usually do not fit into graphics texture memory or even into main memory. Moreover, texturing can be employed to implement visual tools such as magic lenses (see Section 5.1). Therefore, multiresolution modeling for digital terrain models should consider both geometry and texture data.

2 Related Work

Hierarchical triangulations based on TINs have been applied to generate multiresolution models which can be used by level-of-detail (LOD) algorithms (de Floriani et al. [6], Gross et al. [9], Voigtmann et al. [17], and Xia et al. [18]). Regular grids have been used for multiresolution modeling (Falby et al. [8]) and for real-time, continuous LOD rendering (Duchaineau et al. [7], Lindstrom et al. [14], and Pajarola [16]). Hoppe [10] introduced the *view-dependent progressive mesh* which has been further optimized for real-time terrain rendering, and the *geomorph*, a technique to minimize *popping* effects in the terrain representation during changes of the level of detail. Chen et al. [3] discussed a method for combining LOD techniques with *image-based modeling and rendering* techniques to take advantage of the frame-to-frame coherence in screen-space. In many applications the visual quality with respect to topographic terrain features or thematic terrain data is as important as rendering performance. Recent developments (e.g., Hoppe [11], and Xia et al. [18]) take into account the visual quality by considering surface normals, but do not provide an explicit control of the terrain shading as proposed in this paper. Most LOD techniques are limited with respect to the management of large-scale texture data: In contrast to the LOD mechanism for geometry data no similar mechanism is provided for texture data. Lindstrom et al. [15] proposed a method which handles a single large-scale texture related to a LOD terrain geometry. However, an efficient treatment of multiple logical texture layers is required for the interactive exploration and manipulation of terrain data (e.g., terrain visualization used for landscape analysis and planing).

3 Data Structures for Integrated Multiresolution Modeling

We construct a *hybrid terrain model* by a regular grid, called the *reference grid*, and from a collection of TINs, called *microstructures*, which are associated to grid cells and refine the terrain representation within the cell domain. Each of the refining microstructures must adapt itself continuously to the neighboring grid cells and microstructures (see Fig. 1). Grid cells are refined where complex morphology has to be represented (e.g. riverbeds, streets, or ridges). This hybrid terrain model combines the advantages of grids and TINs: it leads to a memory-efficient and morphologically precise terrain representation. Handling the details as precisely as possible is important because we perceive a terrain model mainly by the terrain shading and terrain silhouette, and both depend on geometric details.

3.1 Generic Multiresolution Data Structure for Terrain Geometry

This section defines a multiresolution model for geometry data, the *approximation tree*, which is generic with respect to the type of terrain data as required by hybrid terrain models. Let $P = \{p_1,...,p_n\}$ be a set of n data points in the xy-plane. Let $\mathrm{D}(P)$ be the minimal axis-parallel bounding box of set P, and let $G = (P,h_G)$ be a terrain model defined by the point set P and an elevation function $h_G : \mathrm{D}(P) \rightarrow \mathrm{R}$, which calculates

elevation values for points of D(P) by interpolating the height values for data points of P. The domain of G is defined as D(G) = D(P).

An *approximation tree* $A_{s,d}(G)$ for a terrain model G is represented by a tree; its nodes are called *geometry patches*. Each geometry patch N represents a rectangular region D(N) $\subseteq$ D(G) and approximates the terrain surface G in that region by an approximating terrain surface G(N) = (P(N), $h_{G(N)}$). The set P(N) consists of at most s data points: $|P(N)| \leq s$. Furthermore, the four corner points of D(N) must be contained in P(N). The way the node calculates the data points P(N) depends on the *approximation strategy* adopted by the node. For example, a grid-based node will select evenly spaced points from a grid data set, whereas a TIN-based node will select points from an arbitrary data set based on an error criterion.

The *geometric approximation error* $\varepsilon(N)$ of a geometry patch N is defined by the maximal vertical distance between the terrain models G(N) and G: $\varepsilon(N) = \max_{p \in D(N)} |h_{G(N)}(p) - h_G(p)|$.

Each terrain patch N can have at most d child nodes. The child nodes are constructed as follows: If the geometric approximation error $\varepsilon(N)$ exceeds a certain threshold $\varepsilon \geq 0$, the domain D(N) is decomposed into a set of at most d rectangular, disjoint subdomains D(N_i). The strategy for decomposing a patch depends on the type of the node: a grid-based node applies a quadtree-like subdivision, whereas a TIN-based node is subdivided by a line parallel to the x- or y-axis. For each subdomain D(N_i), a child node N_i of N is constructed which approximates the terrain surface in that subdomain. The domain of the root node of the approximation tree $A_{s,d}(G)$ is D(G) covering the whole domain of the terrain G.

3.2 Multiresolution Data Structure for Terrain Textures

Multiresolution modeling for texture data in the context of multiresolution terrain models is motivated by the following observations:

- Visualization applications are likely to use texture data up to several hundreds of megabytes, for example, in cartographic applications. However, graphics hardware imposes constraints on the size of textures. For example, common OpenGL implementations can process textures up to 1024 x 1024 pixels and constrain the actual size to a power of 2, i.e., 2^m x 2^n pixels.
- The selection of a level-of-detail texture depends on the texture approximation

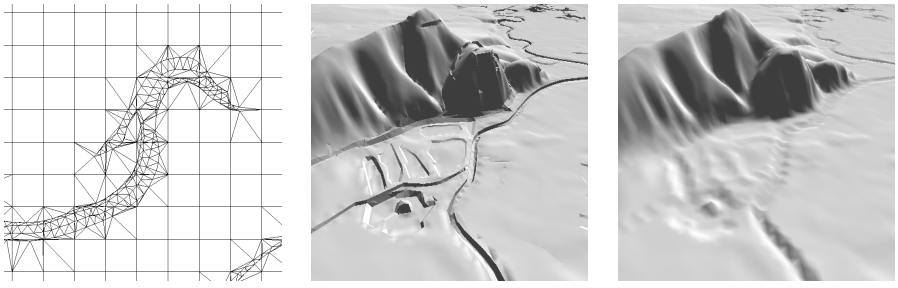

Fig. 1. Hybrid terrain model specified by a reference grid and partially refined by TINs (*left*), view of the terrain model with TINs (*center*) and without TINs (*right*)

error and the patch geometry.

Let T be a geo-referenced 2D texture used for a terrain model G with $D(T) \supseteq D(G)$. The texture pyramid $\Delta(T)$ of a terrain texture T consists of a sequence of textures T_i with decreasing resolution. Conceptually, each texture is created by scaling down the predecessor texture by a factor of ½. The first texture of the sequence is the original terrain texture, the last texture consists of 1 x 1 pixels.

In analogy to the approximation tree for geometric data, we define a similar tree for multiresolution textures, the *approximation tree for terrain textures* $A_{s,d}(G,T)$. The nodes of a tree $A_{s,d}(G,T)$ are called *texture patches*. A texture patch M has the following properties:

- M is associated with exactly one geometry patch $N_M \in A_{s,d}(G)$.
- The domain of M covers the domain of N_M: $D(M) \supseteq D(N_M)$.
- M references that image part S_M with the highest resolution in $\Delta(T)$ which completely covers the domain $D(N_M)$ and fulfills the constraints of the rendering system.
- If S_M is not an image part of the first texture in the texture pyramid $\Delta(T)$ (i.e., it is not part of the original terrain texture T) and the geometry patch N_M has child nodes, then the texture patch M has child nodes, too.
- If the texture patch M has child nodes, then M and N_M have the same number of child nodes, and the domain of a child of M is equal to the domain of the corresponding child of N_M.

The texture resolution of M is considered optimal for the geometry patch N_M. But the geometry patch can also be rendered with any parent texture patch M' of M because its domain covers the domain of M and therefore of N_M, too. In such a case, the texture resolution is non-optimal for the geometry patch, but if the geometry patch is far away from the viewer this reduction of resolution is not visible and can speed up rendering significantly, since less texture data have to be processed.

The rendering algorithm traverses an approximation tree recursively, calculates visual approximation errors, and selects geometry patches and texture patches based on user-defined quality criteria.

3.3 Visual Approximation Errors

Let $A_{s,d}(G)$ and $A_{s,d}(G,T)$ be a geometry approximation tree and an associated texture approximation tree. Let $N \in A_{s,d}(G)$ be a geometry patch with an approximating terrain surface $G(N)$, and let $M \in A_{s,d}(G,T)$ be a texture patch, $D(M) \supseteq D(N)$. Let $B(N)$ be the minimal 3D axis-parallel bounding box of $G(N)$.

The visual approximation errors are defined if the bounding box $B(N)$ intersects the current view volume. If the bounding box intersects, then the point p of the bounding box closest to the camera and inside the view volume can be determined. The visual approximation errors are calculated as follows:

- *Visual geometry approximation error* $\alpha(N)$: Construct a line segment centered at p, parallel to the z-axis (the direction of elevation) having length $\varepsilon(N)$ where $\varepsilon(N)$ denotes the geometric approximation error. Projecting that segment onto the view

plane, the visual geometry approximation error $\alpha(N)$ is the length of the projected line segment measured in pixels (see Fig. 2).

- *Visual texture approximation error* $\gamma(M,N)$: Determine the width w and height h (in the terrain coordinate system) of a texel of the texture patch M. Construct two line-segments centered at p: one is of length w parallel to the x-axis, the other is of length h parallel to the y-axis. Projecting both onto the view plane, the visual texture approximation error $\gamma(M,N)$ is the maximum length of both projected segments measured in pixels.

The visual approximation errors $\alpha(N)$ and $\gamma(M,N)$ are a measure for the visual quality of a geometry patch N and a texture patch M. If we render the approximating terrain surface $G(N)$, it is ensured that each of the pixels of $G(N)$ differs by at most $\alpha(N)$ pixels compared to the original terrain surface G. In analogy, for the texels of the approximating terrain texture M, we can expect that the texels are sufficiently dense for the actual camera settings.

3.4 Recursive Rendering of Approximation Trees

A hybrid terrain model G represented by a tree $A_{s,d}(G)$ together with a terrain texture T represented by a tree $A_{s,d}(G,T)$ is rendered recursively, starting with the pair of root nodes (N_0,M_0). For a pair of nodes (N,M), the algorithm works as follows:

1. If the bounding box $B(N)$ of the current geometry patch does not intersect the current view volume, the recursive rendering stops at this point (*hierarchical view-volume culling*).
2. Otherwise, the visual approximation errors $\alpha(N)$ and $\gamma(M,N)$ are calculated. If $\alpha(N)$ is larger than a user-defined geometric threshold or if $\gamma(M,N)$ is larger than a user-defined texture threshold, then the rendering calls itself recursively for the child patch pairs (N_i,M_i) (*recursive refinement*). If M has no child texture patches M_i, M is used instead.
3. If both visual errors are ok, assign to M the parent texture patch M' of M until the visual texture approximation error $\gamma(M',N)$ exceeds the user-defined texture threshold. The resolution of the parent's texture is lower but this way we guarantee that the resolution comes close to the user-defined texture threshold (*reduction of texture data*).
4. Render the approximating terrain surface $G(N)$ together with the texture determined in the previous step, and terminates the recursion (*rendering*).

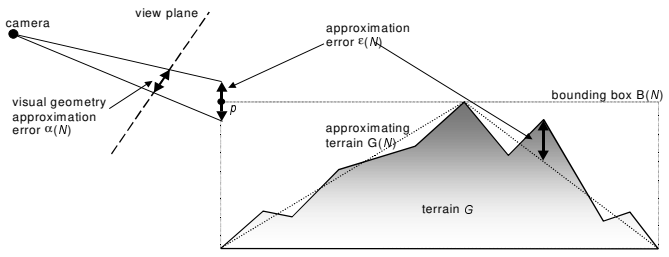

Fig. 2. Visual geometry approximation error $\alpha(N)$ and its calculation

The thresholds for both errors allow the user to prioritize either interactivity or visual quality. Low thresholds lead to higher visual quality, higher thresholds accelerate rendering. Step 3 of the algorithm ensures that the selected texture patch matches the resolution needed for projecting it onto the screen without a visual distortion, but with as few texture data to be processed as possible.

Gaps between two adjacent geometry patches are possible. The gaps are at most as big as the user-defined geometry approximation error. To close the gaps, walls are inserted between geometry patches of different levels of detail. The walls have the same texture coordinates as the edge between the two geometry patches, so they get colored the same way and are visually not recognizable for a reasonable visual geometry approximation error threshold (commonly smaller than 4 pixels).

Only a part of the texture pyramid is actually required to render a single frame. Therefore, texture patches load their texture data on demand, spanning a separate thread. While the texture data is being loaded, the texture data of the parent texture patch can be used. In this case, texture resolution is not optimal, but interactivity is ensured because the application is not blocked and can use at least a reasonable approximation of the required texture. Furthermore, we make use of a mechanism called *file-to-memory mapping* provided by modern operating systems. Each image of the texture pyramid is kept on secondary storage and is only mapped to memory.

4 Shading by Topographic Textures

In many terrain visualization systems the visual quality depends directly on the geometric resolution of the approximating terrain due to the underlying *Gouraud shading*. The vertex normals of a triangle determine the shading of the whole triangle, leading to shading artifacts if triangles become large or thin. For a level-of-detail terrain model this implies that the lower the resolution, the more topographic details get lost, and that shading changes if the LOD-dependent geometry changes. In our approach, the terrain shading relies on *topographic textures*.

A *topographic texture* is precalculated and applied as regular terrain texture to the terrain model, reintroducing topographic details that might be removed during the geometric simplification process. A similar approach has been made in the context of *appearance-preserving simplification strategies* (e.g., Cignoni et al. [4], and Cohen et al. [5]). Terrain models shaded by topographic textures have the following properties:

1. The visual quality of a terrain model depends on the quality of the topographic textures because they encode visually the terrain's morphology.
2. The geometric complexity needed to achieve high-quality images of terrain models is considerably less compared to Gouraud shading because topographic textures are applied pixel-precisely (see Fig. 3).
3. The visual effects of LOD changes can be minimized by topographic textures because the constant shading *hides* discontinuities in the geometric representation during a change in the LOD.

A topographic texture consists of luminance values and depends on the high-resolution terrain model, the terrain surface properties, the lighting conditions, and on special design rules. In cartography, for example, design rules have been developed

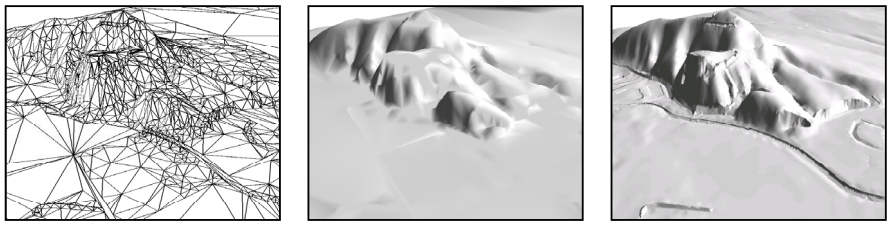

Fig. 3. Wire-frame representation of an approximating terrain model (*left*), Gouraud-shaded model (*center*), shaded by a topographic texture (*right*)

which improves the perception of morphologically important terrain parts such as peaks, pits, valleys and ridges. Note that the ambient and diffuse light, which are the most important ones for terrain shading, do not depend on the camera settings which justifies the precalculation of a topographic texture.

Furthermore, we are not limited to the lighting and shading models provided by the underlying 3D rendering system. Fig. 4 (*left*) shows an automatically generated topographic texture which takes into account self-shadowing of a terrain model. Fig. 4 (*right*) shows a topographic texture calculated for the Himalaya Mountains. The shades and lighting conditions are chosen in such a way that the morphology can be perceived easily.

5 Applications of Multiple Textures

In many applications it is necessary to map two or more thematic textures onto a terrain surface (e.g., topographic texture, road map, and land use information). The approximation tree has been extended to handle more than one texture tree, i.e., a geometry patch can be associated with texture patches of several different texture trees. As a consequence, information layers can recalculate their texture data without affecting the textures of the other layers.

It is important that each texture remains independent: textures of several information layers cannot be merged into one final 2D texture due to their different domains

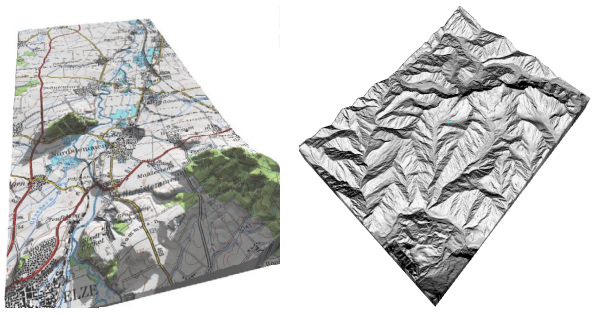

Fig. 4. Topographic texture with self-shadowing and combined with a cartographic texture (*left*), topographic texture based on cartographic shading rules applied to a terrain model of the Himalaya (*right*)

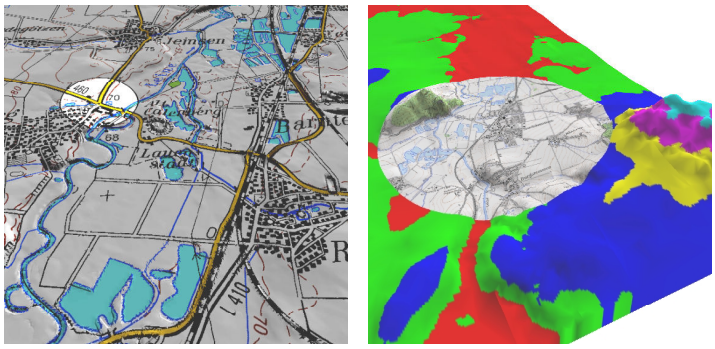

Fig. 5. Applications of multiple textures in terrain visualization: Interactive highlight lens (*left*), thematic lens adding information to a local area (*right*)

and resolutions. In addition, it is not feasible to merge large-scale textures with several hundreds of megabytes in real-time. Furthermore, multiple textures facilitate the implementation of dynamic textures and texture-based animations.

5.1 Texture Lenses and Texture Animations

In Fig. 5 (*left*) an additional luminance texture is used to implement a highlight lens which is combined with a cartographic and a topographic texture. The lens can be used, for example, to control the visual focus of an observer during a presentation. Due to the use of multiple textures the luminance texture needs not to be merged with the high-resolution cartographic and topographic textures and therefore can be moved across the terrain in real-time.

In Fig. 5 (*right*) a thematic lens exhibits a cartographic texture inside a circular region surrounded by a high-contrast height texture which visualizes discrete height regions, everything combined with a topographic shading texture. The visibility of the cartographic texture is restricted by a visibility-restricting texture. Note that visibility-restricting textures normally have low memory requirements because low resolutions (e.g., 128 x 128 pixels) are sufficient.

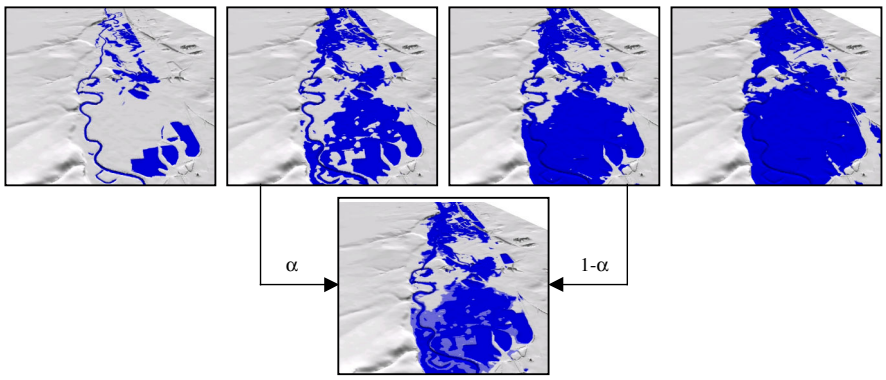

Fig. 6. Key texture frames of a flooding animation (*top*) and an interpolated texture (*bottom*)

To visualize spatio-temporal processes, one can use multiple textures to specify texture key frames. During the animation, we interpolate between two consecutive texture key frames using multitexturing with appropriate weights assigned to both key frame textures. No intermediate texture has to be created which allows us to animate even high-resolution texture sequences. Fig. 6 shows a flooding animation. The texture key frames are 1200 x 2400 pixels large and describe the flooding state in a landscape at concrete time stamps.

5.2 Experiments and Results

All screen shots presented in this paper have been taken from the *LandExplorer* [13], a prototype implementing the described concepts [1]. All time measurements were performed on a standard PC equipped with a 350 MHz Pentium II processor, 128 MB RAM, Riva TNT graphics card with 16 MB graphics memory, and running Windows NT 4.0 (SP6). The window was 640 x 480 pixels large in true-color. We used a terrain data set consisting of about 500.000 triangles (a 640x320 reference grid plus additional fine-structures introducing about 130.000 triangles). The original data set needs more than 5 seconds to render without level-of-detail techniques. The size of the topographic texture is 2500 x 5000 pixels (13 MB, gray-scale) and the thematic texture is 3200 x 6400 pixels (62 MB, RGB) large. The method using Gouraud shading is the slowest method, even rendering with two textures is almost always faster.

6 Conclusions

The hybrid terrain model improves the visual quality of terrain models because microstructures have a great impact on the perception of a terrain model. The rendering process considers both screen-space geometric errors and texture errors which control

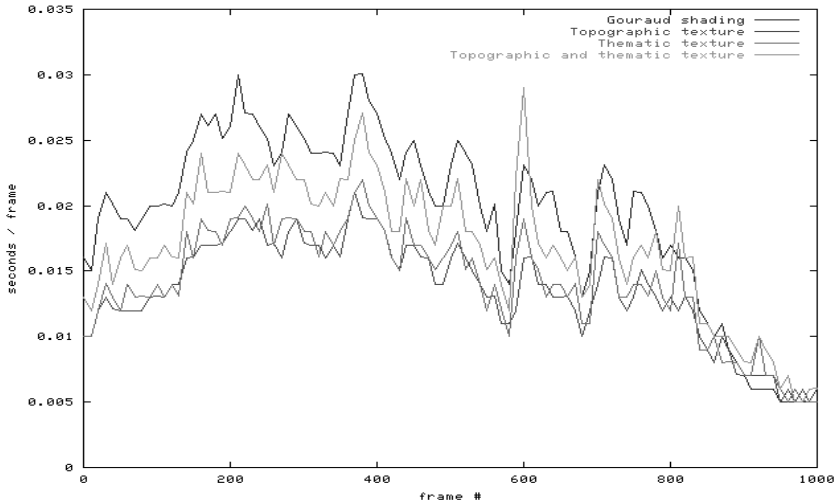

Fig. 7. Timings for different visualization techniques

visual quality and geometric correctness. As a direct application of multiple textures, shading can be implemented by topographic textures improving the visual quality dramatically because they outwit human perception by providing detailed and LOD-independent shading information for a coarse geometry. Using multiple textures proves to be feasible because multitexturing for its implementation is available on modern graphics hardware. Currently, a cartographic visualization system [2] is being developed based on the presented multiresolution model.

References

[1] K. Baumann, J. Döllner, K. Hinrichs, O. Kersting: *A Hybrid, Hierarchical Data Structure for Real-Time Terrain Visualization*. Proc. Computer Graphics Intern. '99, 85-92, 1999.

[2] G. Buziek, J. Döllner: *Concept and Implementation of an Interactive, Cartographic Virtual Reality System*. Proceedings of the International Cartographic Conference ICC '99, Ottawa, 637-648, 1999.

[3] B. Chen, J. Swan, E. Kuo, A. Kaufman: *LOD-Sprite Technique for Accelerated Terrain Rendering*. Proceedings IEEE Visualization '99, 1999.

[4] P. Cignoni, C. Montani, C. Rocchini, R. Scopigno: *A general method for preserving attribute values on simplified meshes*. Proceedings IEEE Visualization '98, 59-66, 1998.

[5] J. Cohen, M. Olano, D. Manocha: *Appearance-Preserving Simplification*. Proceedings of SIGGRAPH '98, 115-122, 1998.

[6] L. De Floriani, P. Magillo, E. Puppo: *Efficient Implementation of Multi-Triangulations*. Proceedings IEEE Visualization '98, 1998.

[7] M. Duchaineau, M. Wolinsky, D. Sigeti, M. Miller, C. Aldrich, M. Mineev-Weinstein: *ROAMing Terrain: Real-time Optimally Adapting Meshes*. Proceedings IEEE Visualization '97, 81-88, 1997.

[8] J. Falby, M. Zyda, D. Pratt, R. Mackey: *NPSNET: Hierarchical Data Structures for Real-Time Three-Dimensional Visual Simulation*. Computers & Graphics, 17(1):65-69, 1993.

[9] M. Gross, R. Gatti, O. Staadt: *Fast Multiresolution Surface Meshing*. Proceedings IEEE Visualization '95, 135-142, 1995.

[10] H. Hoppe: *Smooth View-Dependent Level-of-Detail Control an its Application to Terrain Rendering*. Proceedings IEEE Visualization '98, 35-42, 1998.

[11] H. Hoppe: *New quadric metric for simplifying meshes with appearance attributes*. Proceedings IEEE Visualization '99, 59-66, 1999.

[12] M. Kofler, M. Gervautz, M. Gruber: *The Styria Flyover - LOD management for huge textured terrain models*. Proc. Computer Graphics International '98, 444-454, 1998.

[13] Land*Explorer*. WWW-Site: `http://www.mamvrs.de/geovisualiz.htm`, 1999.

[14] P. Lindstrom, D. Koller, W. Ribarsky, L. Hodges, N. Faust, G. Turner: *Real-Time, Continuous Level of Detail Rendering of Height Fields*. Proc. SIGGRAPH '96, 109-118, 1996.

[15] P. Lindstrom, D. Koller, L. Hodges, W. Ribarsky, N. Faust, G. Turner: *Level-of-detail Management for Real-Time Rendering of Phototextured Terrain*. GVU TR 95-06, 1995.

[16] R. Pajarola: *Large Scale Terrain Visualization using the Restricted Quadtree Triangulation*. Proceedings IEEE Visualization '98, 19-26, 1998.

[17] A. Voigtmann, L. Becker, K. Hinrichs: *A Hierarchical Model for Multiresolution Surface Reconstruction*. Graphical Models and Image Processing, 59:333-348, 1997.

[18] J. Xia, J. El-Sana, A. Varshney: *Adaptive Real-Time Level-of-detail-based Rendering for Polygonal Models*. IEEE Transactions on Visualization and Computer Graphics '97, 3(2):171-183, 1997.